Q1] Write a java application to implement interfaces

Step1:- Start

step2:- Create object Demo d1 = new Demo()

step3:- call d1.display()

step4:- Call d1.show()

step5:- Stop

display ()

step 1:- d1.display()

step 2:- Display I am in interface

step3:- Return

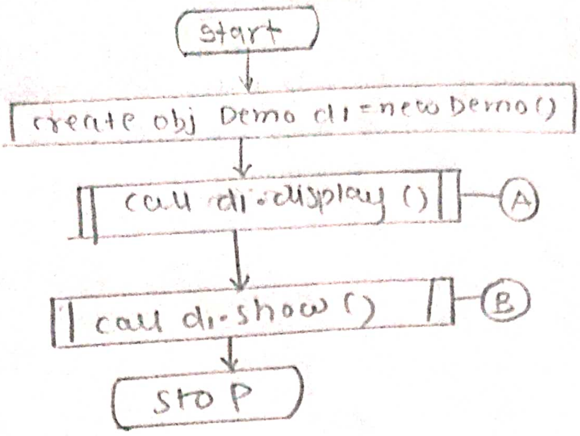
show ()

step 1:- d1.show()

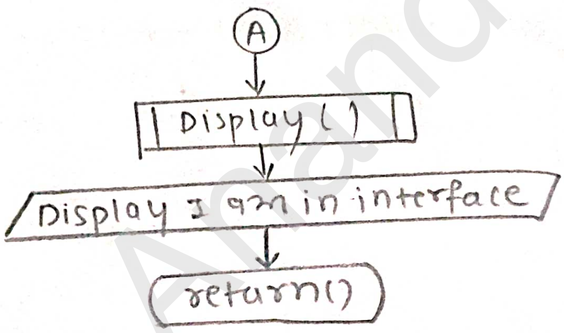
Step 2:- Display "x ="+x

Step 3:- Return

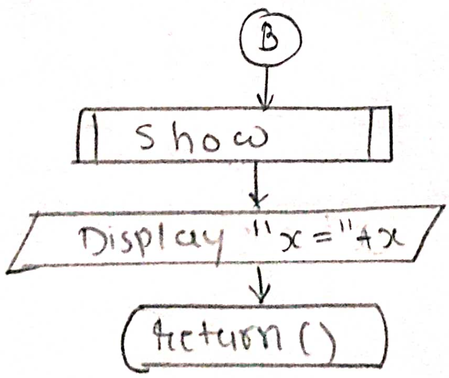
1] Flowchart :-



* function display ()



* function show()



Q2] write a java application to implement abstract class

Algorithm :-

step1 :- start

step2 :- create object Demo d_1 = new Demo()

step3 :- call d_1.display()

step4 :- call d_1.show()

step5 :- stop

Display()

step1 :- call d_1.display()

step2 :- Display I am in abstract class

step3 :- Return

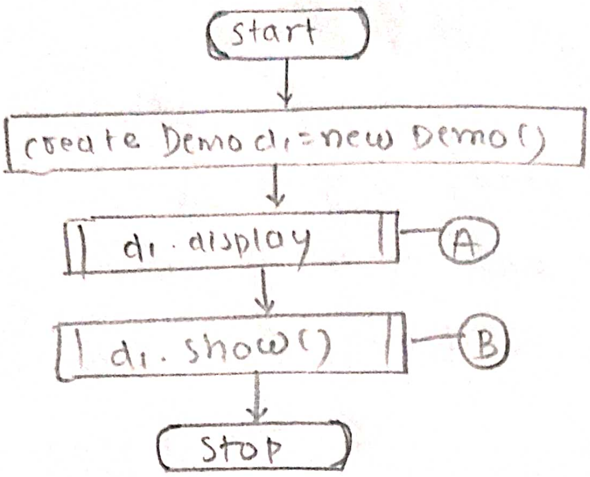
show()

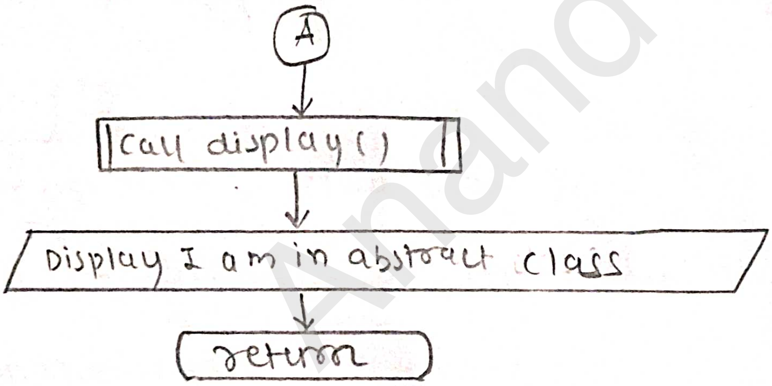
step1 :- call d_1.show()

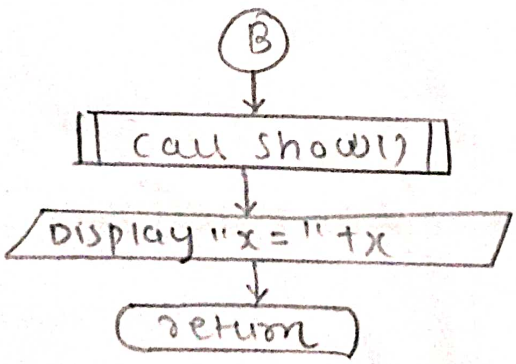
step2 :- Display "x =" x

step3 :- return

Q2] Flowchart :-







Q3] write a java application to implement final class

Algorithm :-

step 1 :- start

Step 2 :- create object A a1 = new A()

step 3 :- create object Demo2 d2 = new Demo2()

step 4 :- call a1. display()

step 5 :- call d2 . show()

step 6 :- stop

Display ()

step 1 :- call a1. display()

Step 2 :- Display I am in final class

Step 3 :- return

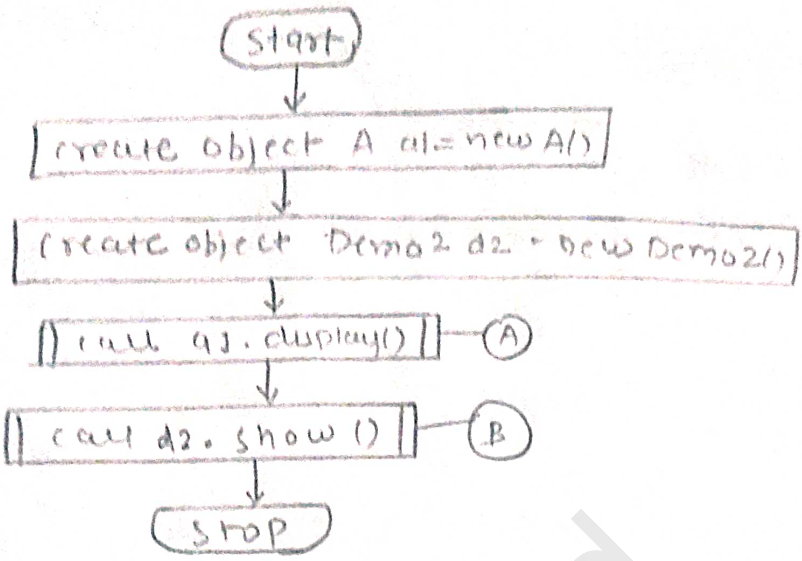
show ()

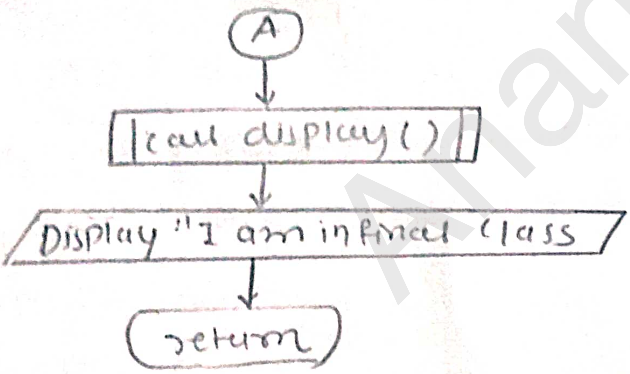
Step 1 :- call d2. show()

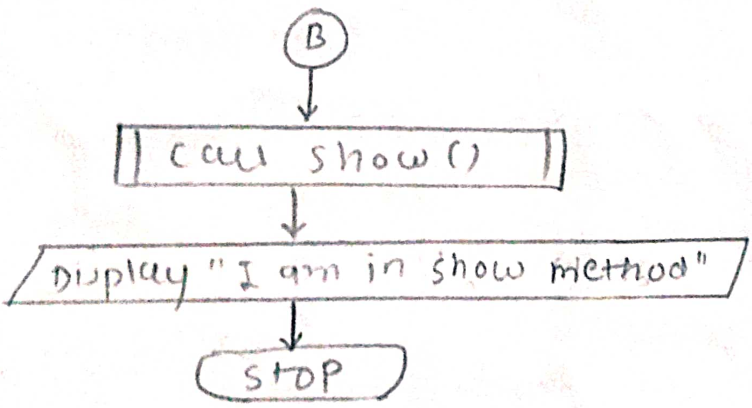
Step 2 :- Display I am in show method

step 3 :- return

(B) Flowchart


Start
Create object A a1 = new A()
Create object Demo2 d2 = new Demo2()
call a1. display()
A
call d2. show ()
B
stop


A
call display()
Display "I am in final class
return


B
call show ()
Display "I am in show method"
STOP

Q4] Write a java application to implement multithreading

Algorithm :-

step 1 :- start

step 2 :- create a class ThreadDemo

step 3 :- create object new thread n_1 = NewThread("one")

step 4 :- create object newthread n_2 = New Thread ("two")

step 5 :- create object newthread n_3 = NewThread ("three")

step 6 :- for int i = 1

step 7 :- i <= 5

step 8 :- Display "main =" + i

step 9 :- try

step 10 :- Thread.sleep(500)

step 11 :- catch (Exception e)

step 12 :- Display e

step 13 :- i = i + 1

step 14 :- Stop

Run ()

step 1 :- call Run ()

step 2 :- for int i = 1

step 3 :- i <= 5

step 4 :- Display ' getName() + " = " + i '

step 5 :- try

step 6 :- Thread . sleep (500)

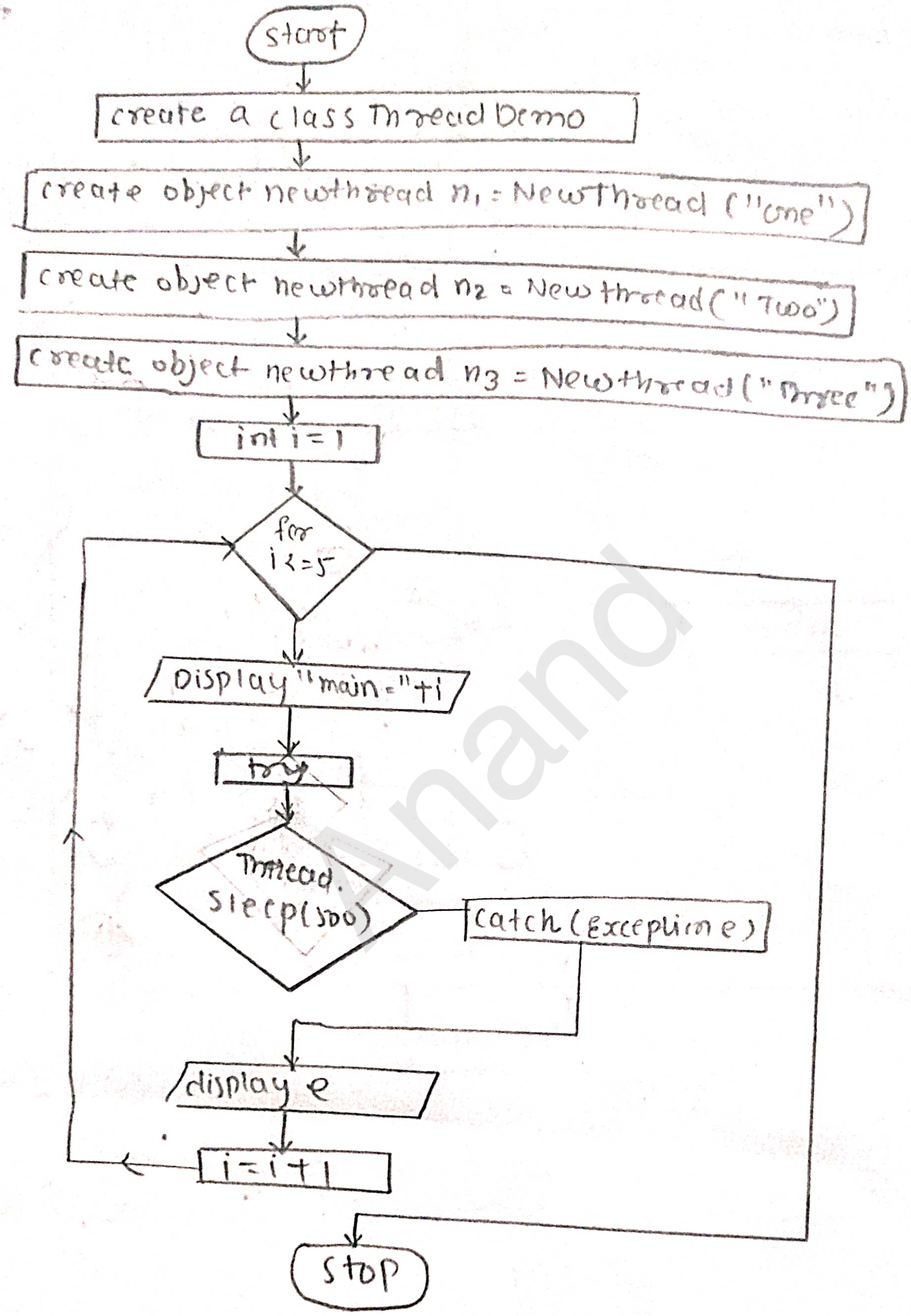
step 7 :- catch . (Exceptione)

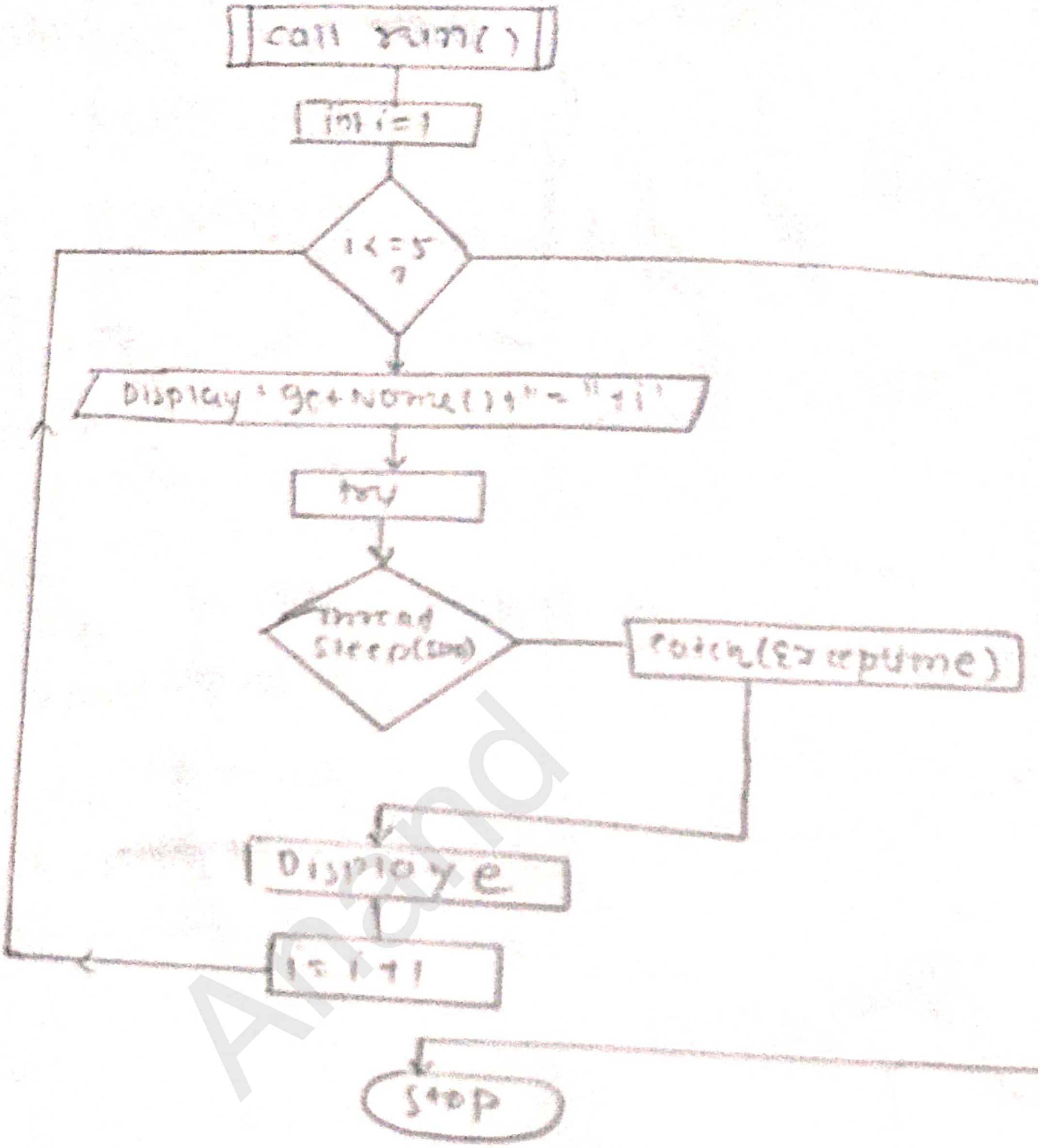
step 8 :- Display (e)

step 9 :- i = i+1

step 10 :- return

Q4] Flowchart



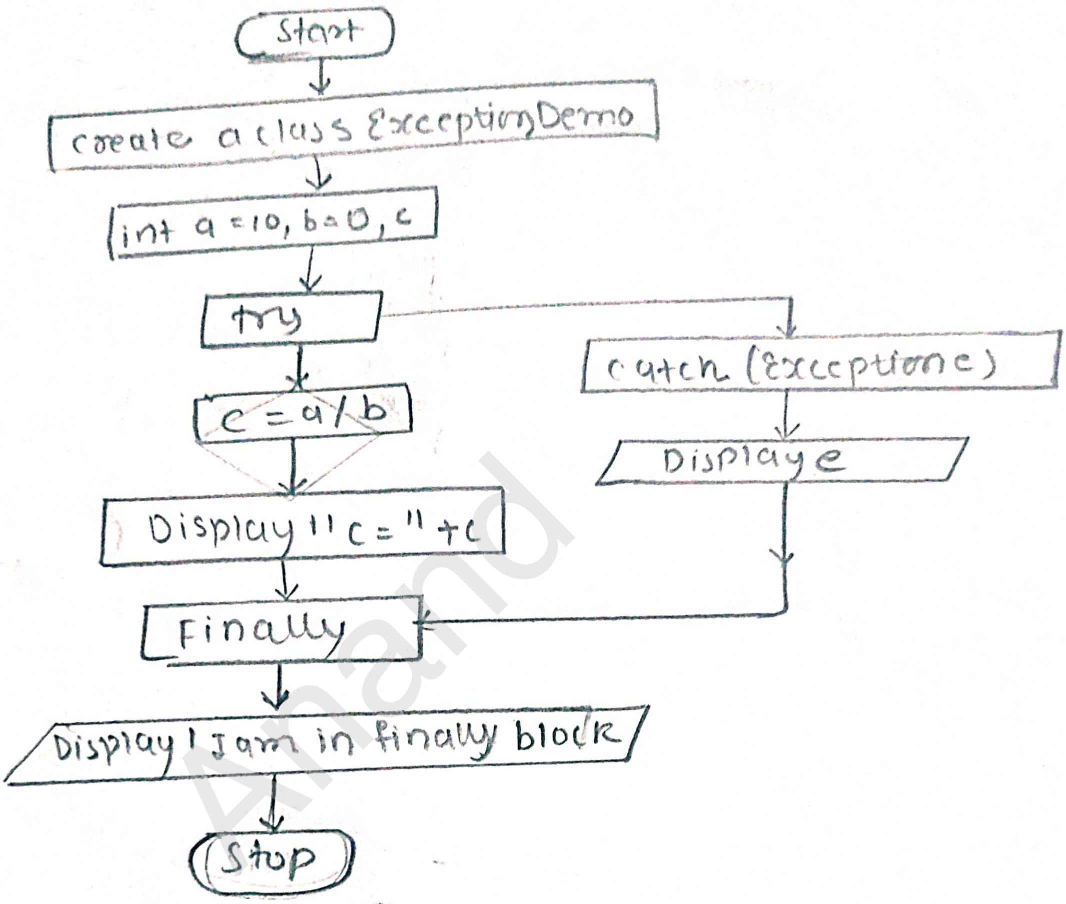

call run()
int i=1
i<=5 ?
try
Display e
stop

Q5] Write a java application to implement exception handling

Algorithm:-

step 1:- Start
step 2:- create a class ExceptionDemo
step 3:- int a=10, b=0, c.
step 4:- try
step 5:- c=a/b
step 6:- Display "c="+c
step 7:- catch (Exception e)
step 8:- Display e
step 9:- Finally
step 10:- Display 'I am in finally block'
step 11:- Stop

Q5] Flowchart :-



Q6] write a java application to implement single level inheritance

Algorithm:-

step1 :- start

step2 :- create a class InheritDemo

step3 :- create object B b1 = new B()

step4 :- call b1. display()

step5 :- call b2 . display1()

step6 :- stop

Display()

step1 :- call b1. display

step2 :- Display "x = " + x

step3 :- return

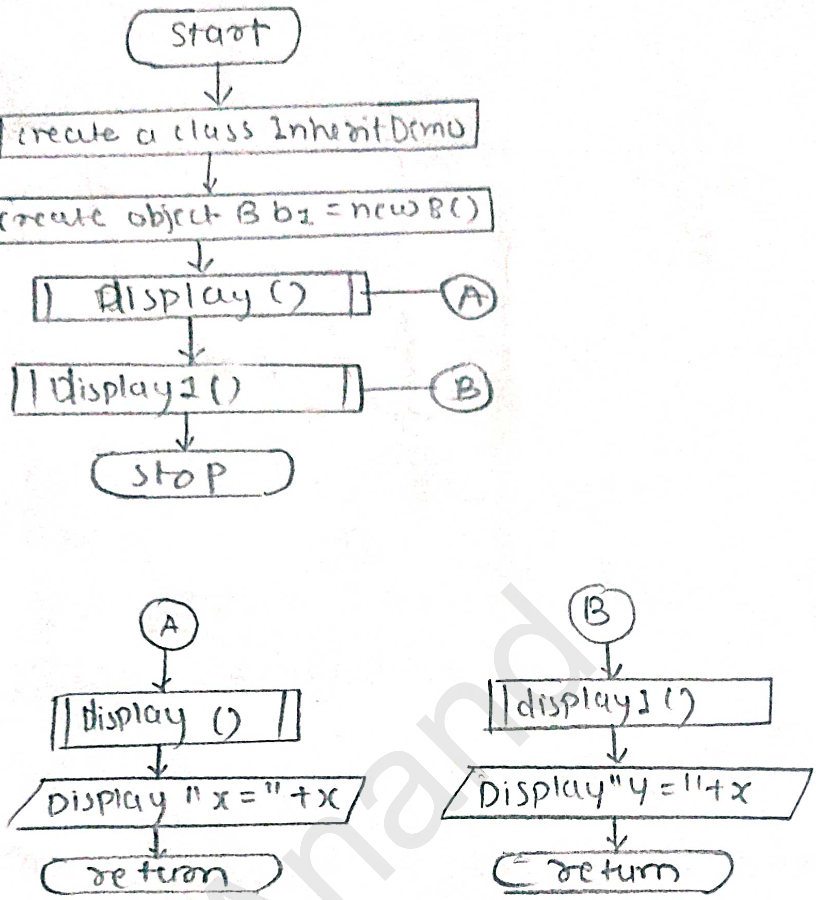
Display1()

step1 :- call b1. display1()

step2 :- Display "y = " + y

step3 :- return

Q6] Flowchart:-



Q7] write a java application to implement passing parameter to Applet

Algorithm:-

step1 :- start

step2 :- user applet for creating passing parameters

step3 :- create graphics object

step4 :- call g for draw string

step5 :- stop

void Init()

step 1 :- void init

step 2 :- str = getParameter ("string")

step 3 :- if (str = null)

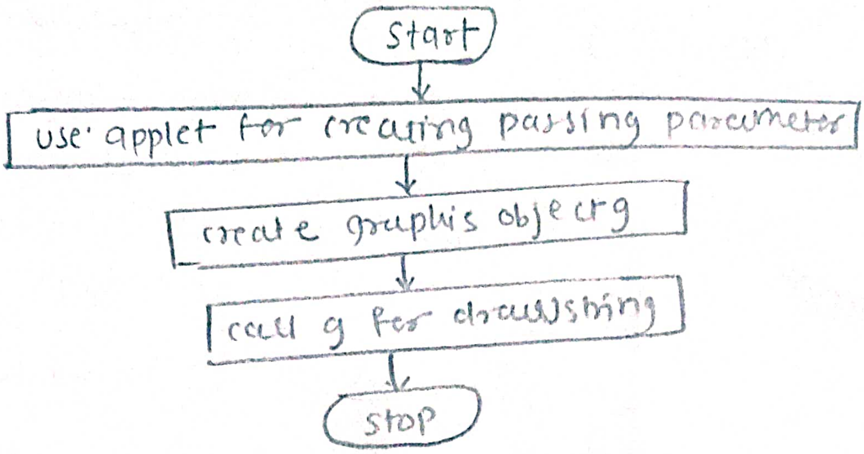
Display "Java"

Else

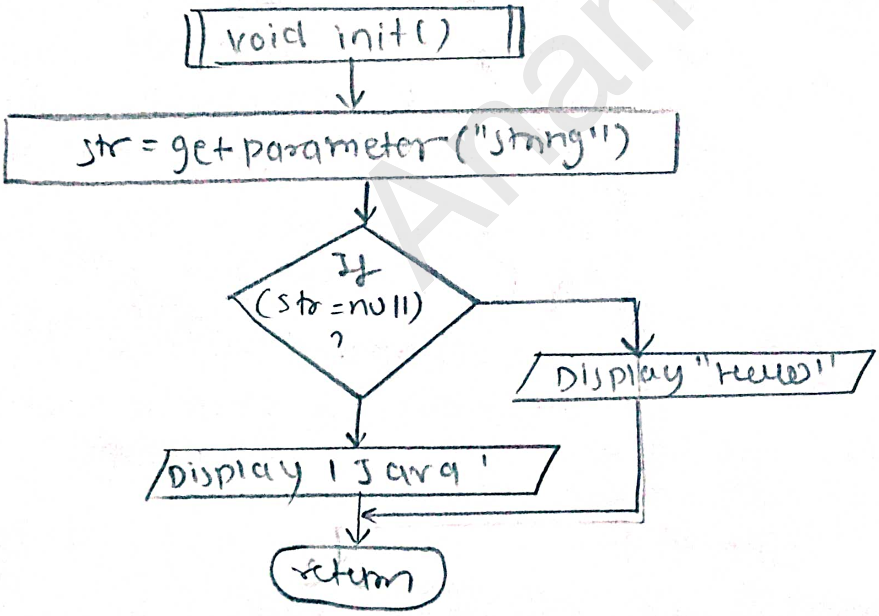
Display " Hello"

step 4 :- return

Q7] Flowchart :-



void init()



Q8] write an AWT based java application for TextArea control

Algorithm:-

step1 :- Start

step2 :- Create object TextArea f = new TextAreaApp()

step3 :- Call f. set Visible (true)

step4 :- Stop

TextAreaAPP()

step1 :- setTitle ("my window")

setsize (400, 400)

set layout (null)

step2 :- t1 = new TextArea()

t1 . setBounds (20, 50, 200, 150)

add (t1)

step3 :- t2 = new TextArea()

t2 . set Bounds (20, 245, 200, 150)

add(t2)

step4 :- b1 = new Button ("copy")

b1. set Bound (20, 210, 70, 25)

add(b1)

step5 :- Call b1. addActionListener(newActionListener()

public void actionPerformed (ActionEvent ae)

t2. setText (t1.getText());

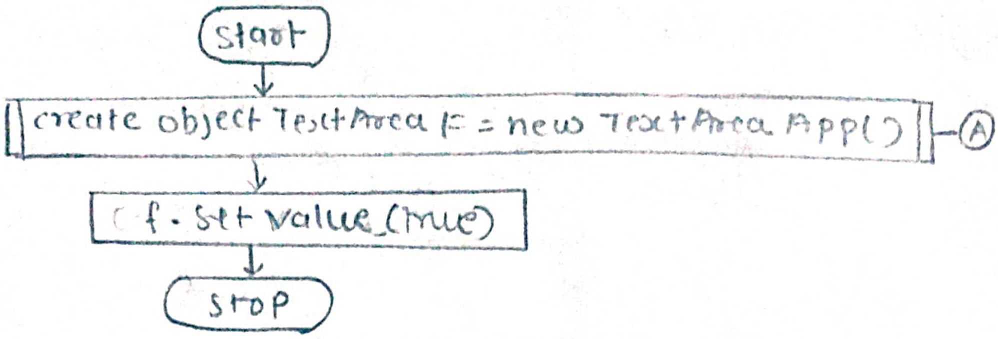
step6 :- call this. addWindowListener(new WindowAdapter()

public void windowClosing (window Event we)

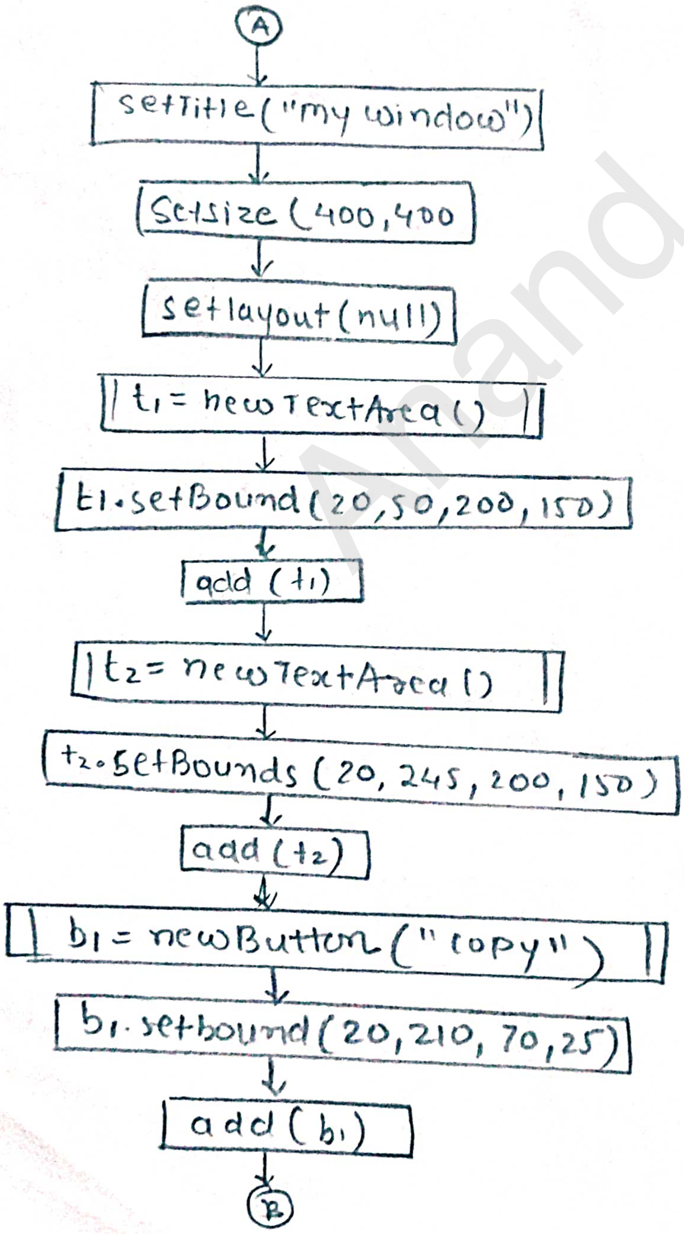
System. exit (0)

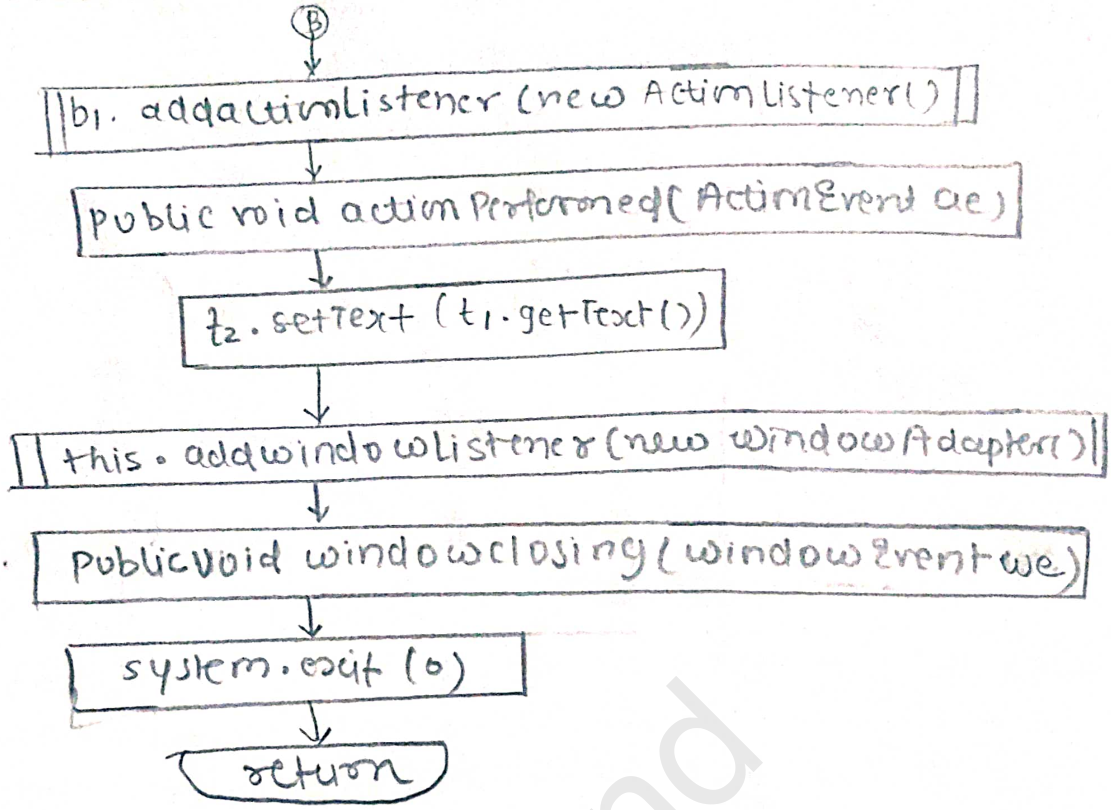
step 7 :- Return

Q8] Flowchart :-



TextAreaApp()




B
b1. addactionlistener (new ActionListener()
public void actionPerformed(ActionEvent ae)
t2. setText (t1. getText())
this. addwindowListener (new windowAdapter()
Publicvoid windowclosing (windowEvent we)
system. exit (0)
return

Q19] write an AWT based Java application to implement ItemEvent of CheckBox control.

Algorithm:-

step1 :- start
step2 :- call new myDemo()
step3 :- stop

MyDemo()

step1 :- call my Demo()
step2 :- create object frame f = new frame ("CheckBox Example")
step3 :- create object label = new label()
step4 :- label.setAlignment(label.CENTER)
label.setSize(400,100)
step5 :- checkBox1 = new checkBox("C++")
checkBox1.setBound(100,50,50,50)
step6 :- CheckBox2 = new checkBox("Java")
CheckBox2.setBounds(100,100,50,50)
step7 :- f.add(checkBox1)
f.add(checkBox2)
f.add(label)
step8 :- f.setSize(400,400)
f.setlayout(null)
f.setVisible(true)
step9 :- call itemStateChanged(ItemEvent e)
step10 :- If(e.getSource()==checkBox1)

Yes :- label. setText ("C ++ CheckBox " + (e.getStateChange() == 1 ? "checked" : "unchecked"));

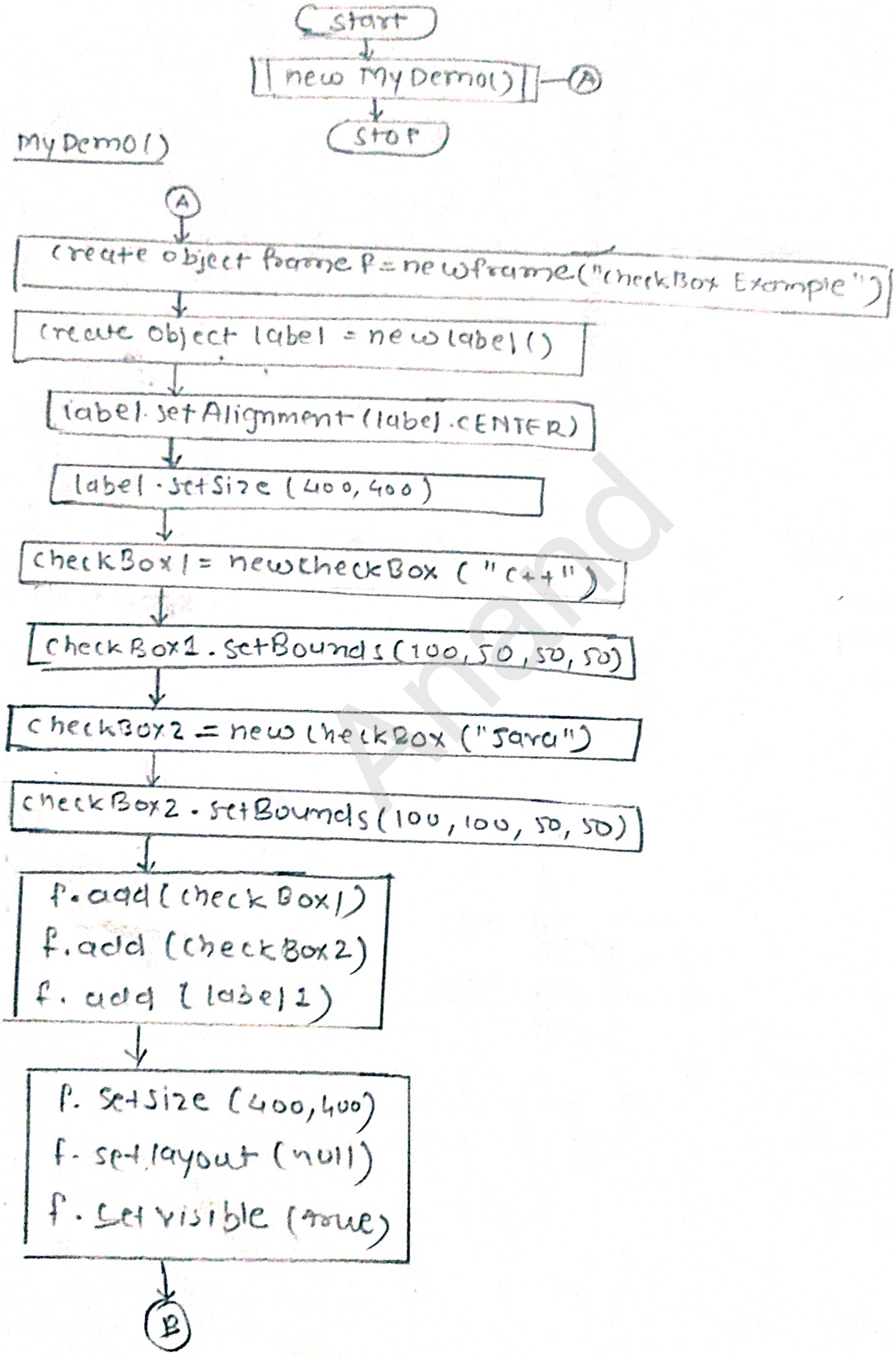
NO :- If (e.getSource() == checkBox2)

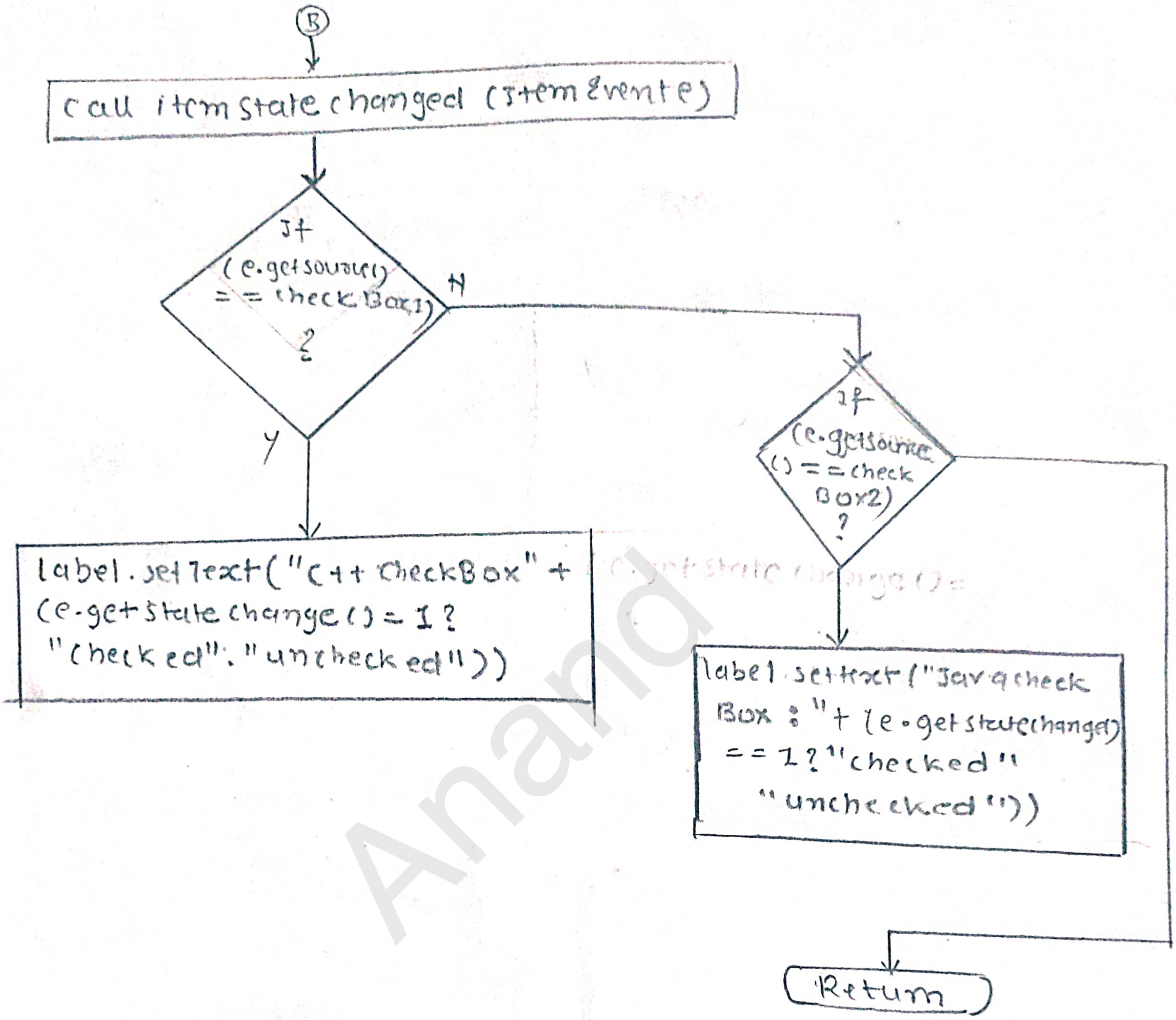
Yes :- label. setText ("Java CheckBox : " + (e.getStateChange() == 1 ? "checked" : "unchecked"));

NO :- Return .

Anand

9] Flowchart


start
new MyDemo()
A
stop
MyDemo()
A
create object frame f = new frame("CheckBox Example")
create object label = new label()
label.setAlignment(label.CENTER)
label.setSize(400,400)
checkBox1 = new checkBox("C++")
checkBox1.setBounds(100,50,50,50)
checkBox2 = new checkBox("Java")
checkBox2.setBounds(100,100,50,50)
f.add(checkBox1)
f.add(checkBox2)
f.add(label1)
f.setSize(400,400)
f.setLayout(null)
f.setVisible(true)
B


B
call item state changed (Item Event e)
If (e.getsource() = = checkBox1) ?
N
Y
label.setText("C++ CheckBox" + (e.getState change() = 1 ? "checked" : "unchecked"))
If (e.getsource() = = check Box2) ?
label.settext("Java check Box :" + (e.getstatechange() = = 1 ? "checked" "unchecked"))
Return

Q10]

Q10] Write a SWING based java application to Implement Click event of Button control.

Algorithm :-

step 1 :- start

Step 2 :- create object
JFrame f = new JFrame ("Button Example")

Step 3 :- create object
final JTextField tf = new JTextField ()

Step 4 :- tf.setBounds (50, 50, 150, 20)

Step 5 :- JButton b = new JButton ("click Here")

Step 6 :- b.setBounds (50, 100, 95, 30)

Step 7 :- call b.addActionListener (new ActionListener ()

step 8 :- call actionPerformed (ActionEvent e)

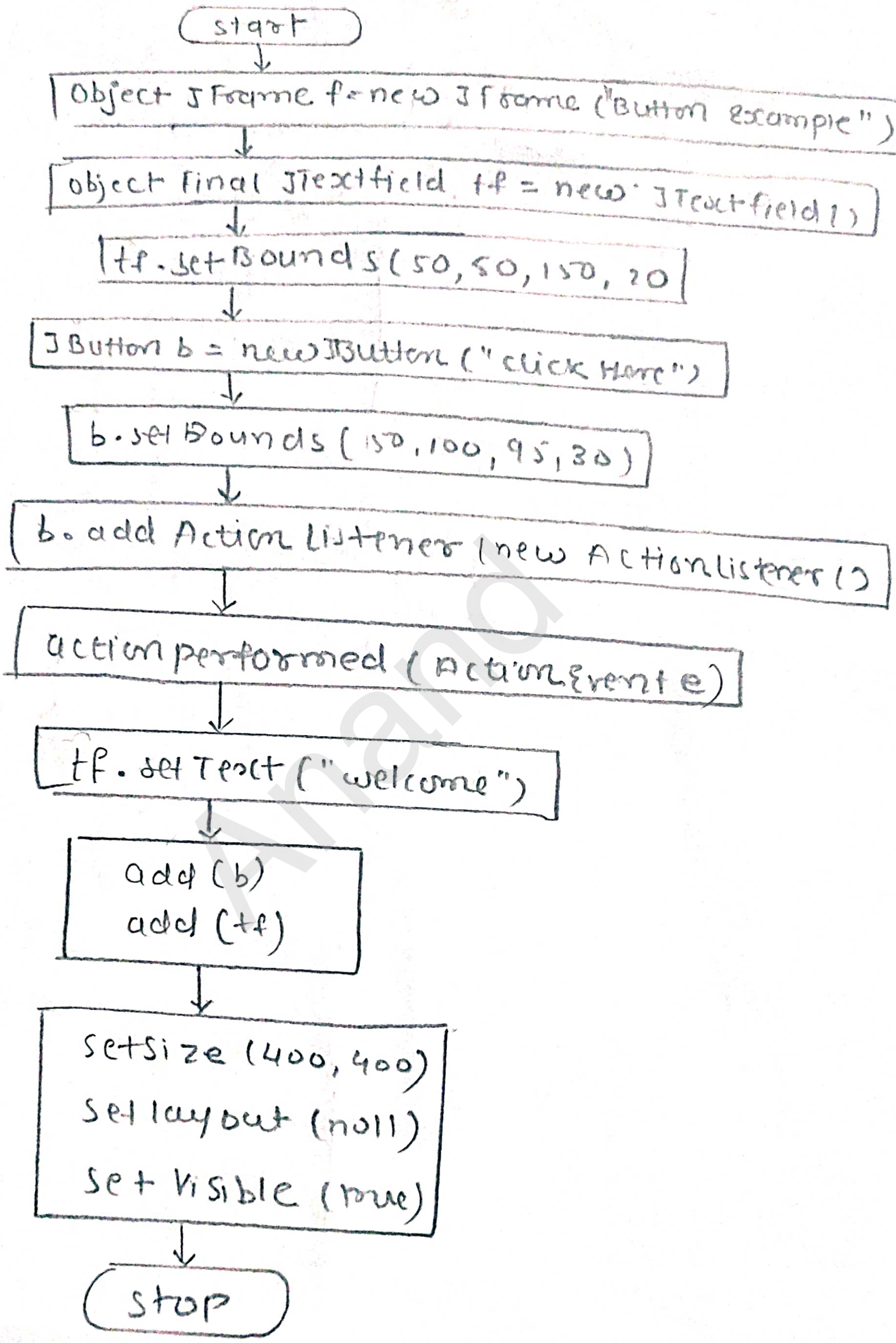
Step 9 :- tf.setText ("welcome")

Step 10 :- add (b)
add (tf)

step 11 :- setSize (400, 400)
Set layout (null)
Set visible (true)

Step 12 :- stop

Q10] Flowchart :-



Q10] Write a java application to create server and client program using TC/IP protocol

Algorithm :-

Server side :-

Step 1 :- Start

Step 2 :- try

Step 3 :- create object ServerSocket S1 = new ServerSocket(420)

Step 4 :- create object socket S2 = S1.accept()

Step 5 :- Input Stream in = S2.getInputStream()

Step 6 :- Int character

Step 7 :- while ((ch=in.read())!=-1)

Display "(char) ch"

Step 8 :- in.close()

Step 9 :- S2.close()

Step 10 :- S1.close()

Step 11 :- Catch (Exception e)

Display(e)

Step 12 :- stop

Client side

step1 :- start

step2 :- try

step3 :- create object s = new socket ("127.0.0.1", 420)

step4 :- outputStream out = s.getOutputStream()

step5 :- set string str = "good morning mca-msc"

step6 :- byte b[] = str.getBytes()

step7 :- out.write (b)

step8 :- out.close ()

step9 :- s.close ()

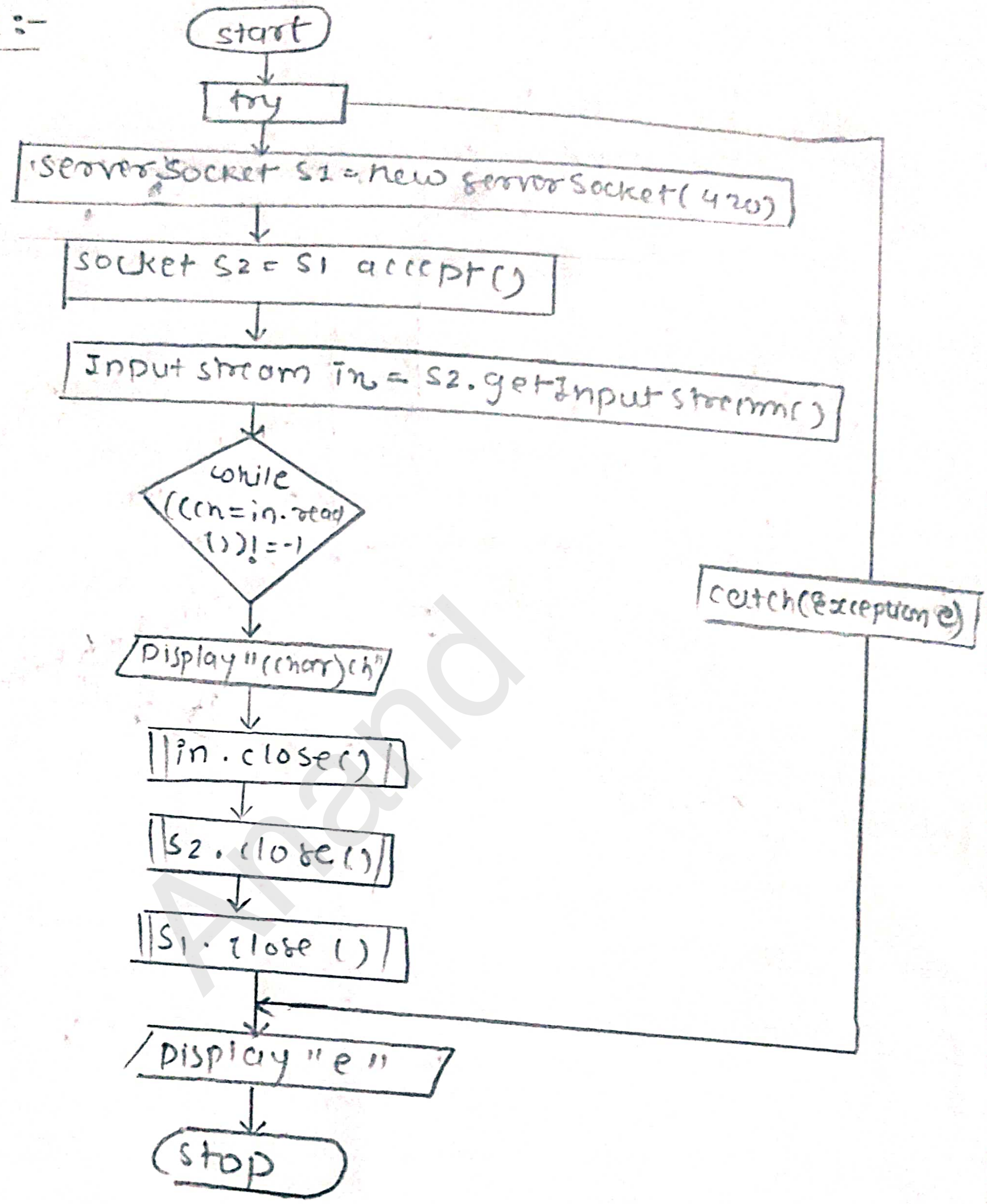
step10 :- catch (Exceptione)

Display "e"

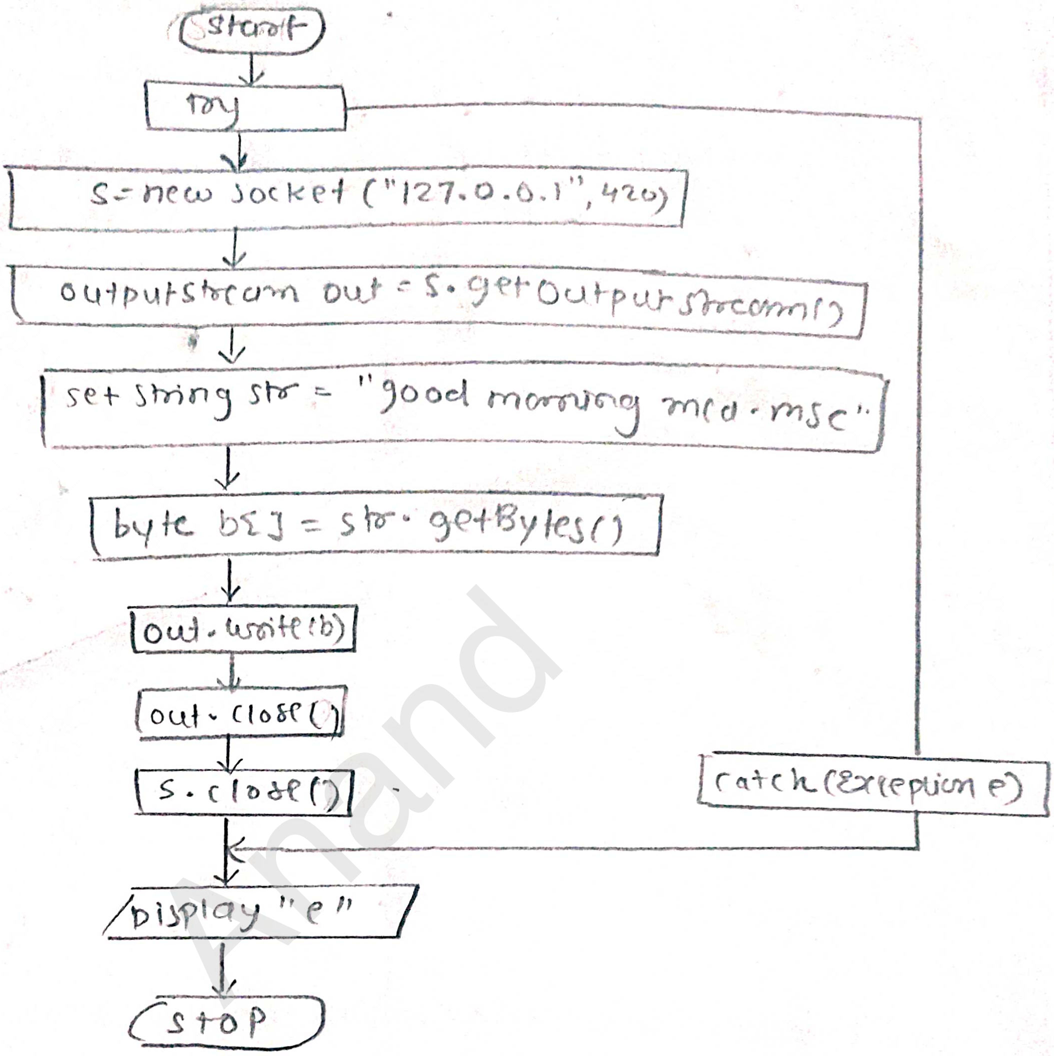
step11 :- stop

Q11] Flowchart:-

server side:-



Client side :-


Start
try
s = new Socket ("127.0.0.1", 420)
outputstream out = s.getOutputStream()
set String str = "good morning mca.msc"
byte b[] = str.getBytes()
out.write(b)
out.close()
s.close()
catch (Exception e)
Display "e"
STOP

12] Write a java application to create server and client program using Datagram

Algorithm :- Sender :-

Step1 :- start

Step2 :- try

Step3 :- create object DatagramSocket ds = new DatagramSocket(101)

Step 4 :- String str = "good morning"

Step5 :- byte b[] = str.getBytes()

Step6 :- create objet DatagramPacket dp = new DatagramPacket (b, b length, InetAddress.getByName("localhost"), 102)

Step7 :- call ds.send (dp)

Step8 :- call ds.close()

Step9 :- catch (Exception e)
Display "e".

Step10 :- stop

Receiver :-

Step1 :- Start

Step2 :- try

Step3 :- create obj DatagramSocket ds = new DatagramSocket(102)

Step4 :- byte b[] = new byte[100]

Step5 :- create obj DatagramPacket dp = new DatagramPacket (b, b.length)

Step6 :- call ds.receive(dp)

Step7 :- display(new String(b))

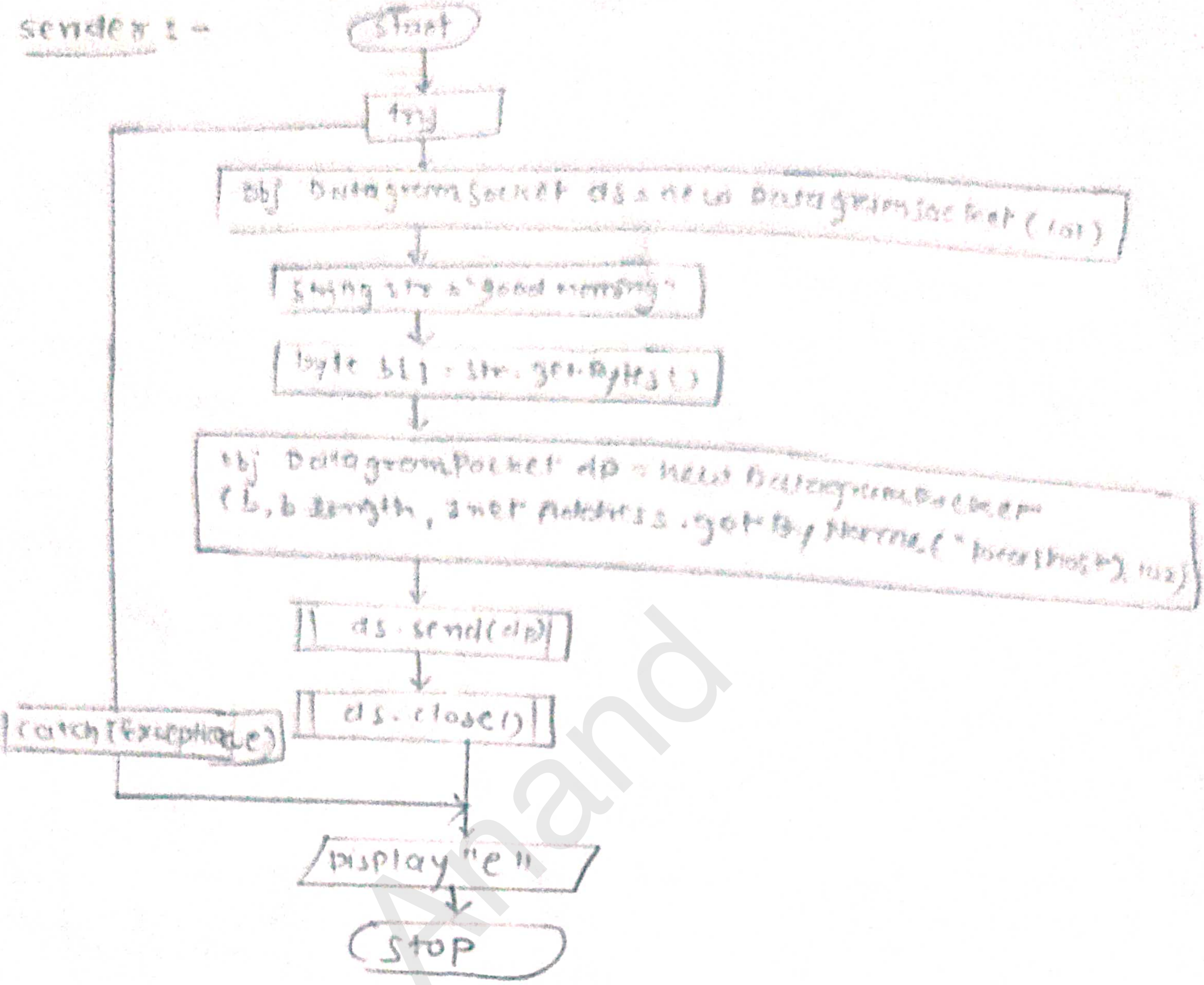
Step8 :- call ds.close()

Step9 :- catch(Exception e)

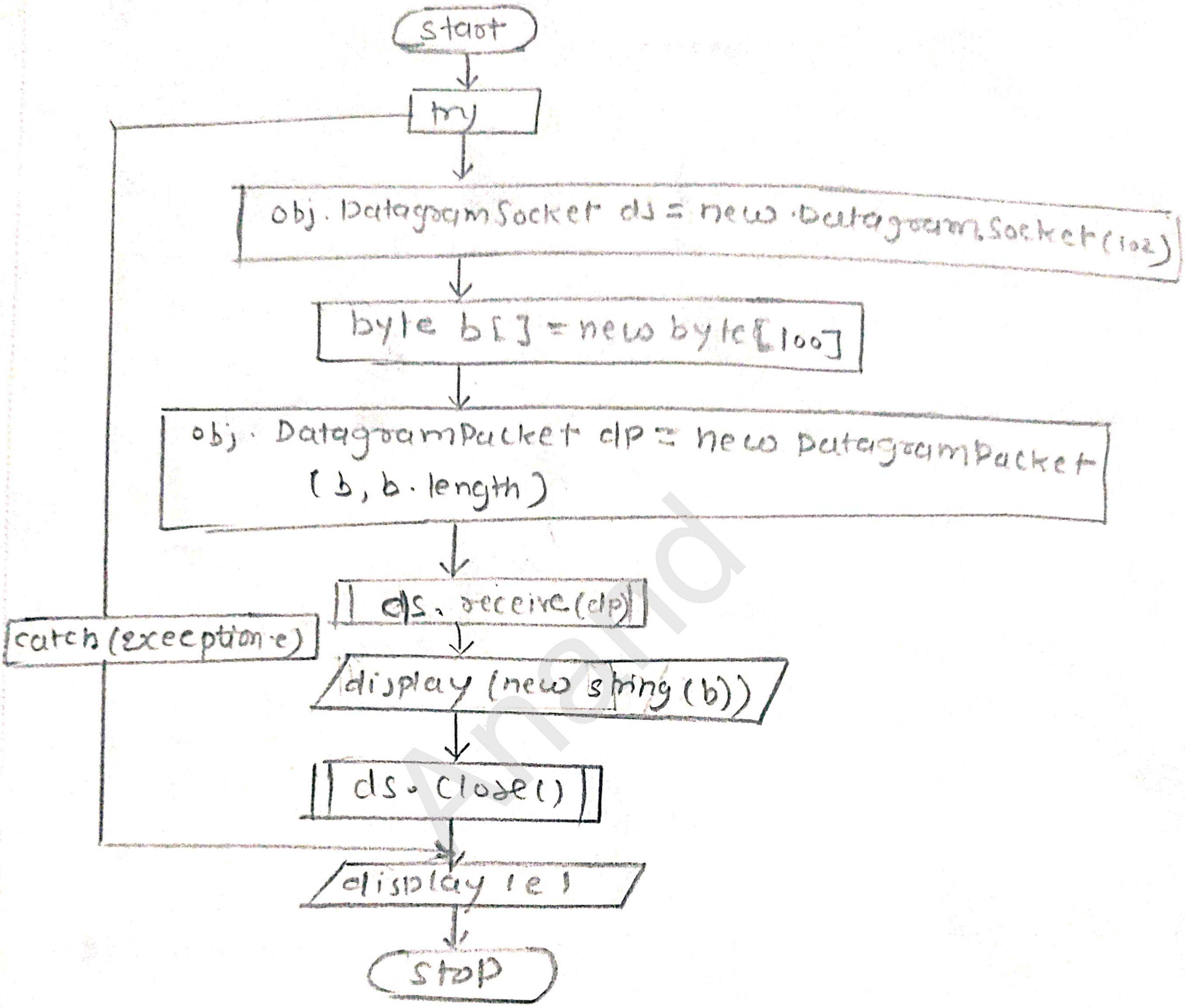
Step 10 :- Display (e)

(b)(c)] Flowchart

sender 1 -



Receiver:-


start
try
obj. DatagramSocket ds = new DatagramSocket(102)
byte b[] = new byte[100]
obj. DatagramPacket dp = new DatagramPacket
(b, b.length)
ds. receive(dp)
catch (exception e)
display (new string (b))
ds. close()
display (e)
stop

Q13] Write a java application to implement RMI program for addition of two numbers.

Algorithm:-

Adder Interface :-

Step1 :- Call interface Adder from Remote

Step2 :- int add (int x, int y)

Step3 :- stop

Adder Implementation class.

Step1 :- Start

Step2 :- Call AdderRemote from UnicastRemoteObject

Step3 :- AdderRemote () throws RemoteException

Step4 :- super()

Step5 :- int add (int x & int y)

Step6 :- return x+y

Step7 :- Stop

Server class :-

step 1 :- start
step 2 :- try
step 3 :- Adder stub = new AdderRemote()
step 3 :- Naming rebind ("hello", stub)
step 4 :- catch (Exception e)
~~Step~~ Display (e)
step 5 :- stop

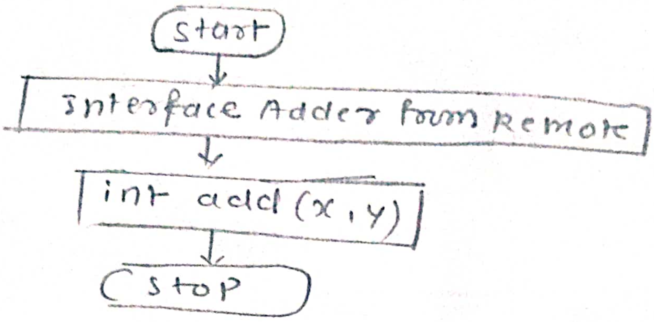
Client class :-

step 1 :- start
step 2 :- try
step 3 :- create object Adder stub = (Adder)
Naming.lookup("hello")
Display (stub. add (10, 20))
step 4 :- catch (Exception e)
~~St~~ Display "e"

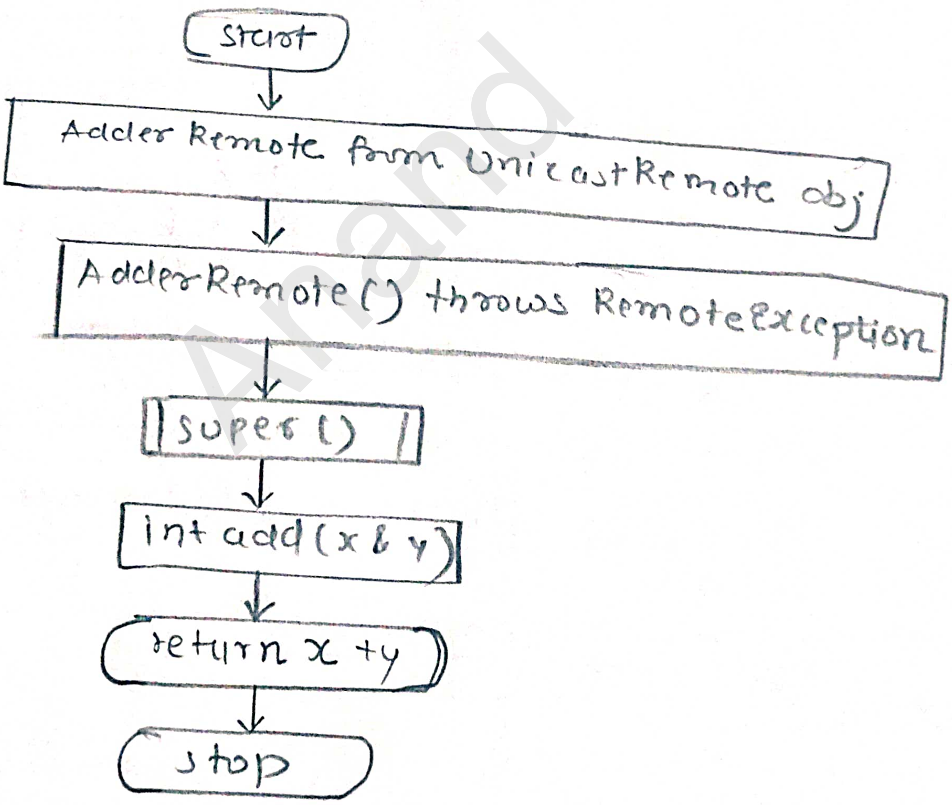
step 5 :- stop

(Q18) Flowchart:-

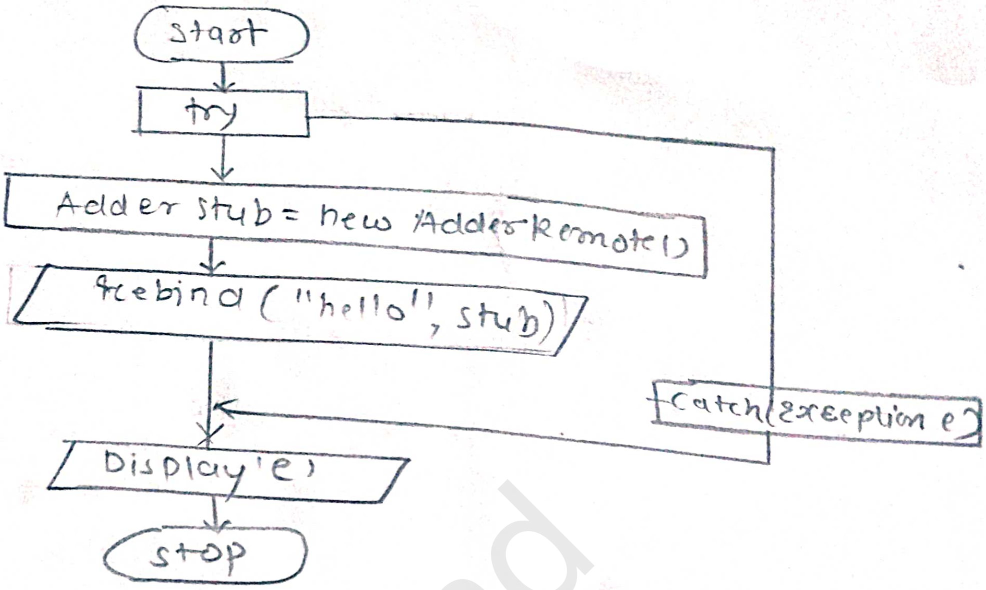
Adder Interface



Adder Implement class



server class



client class :-

